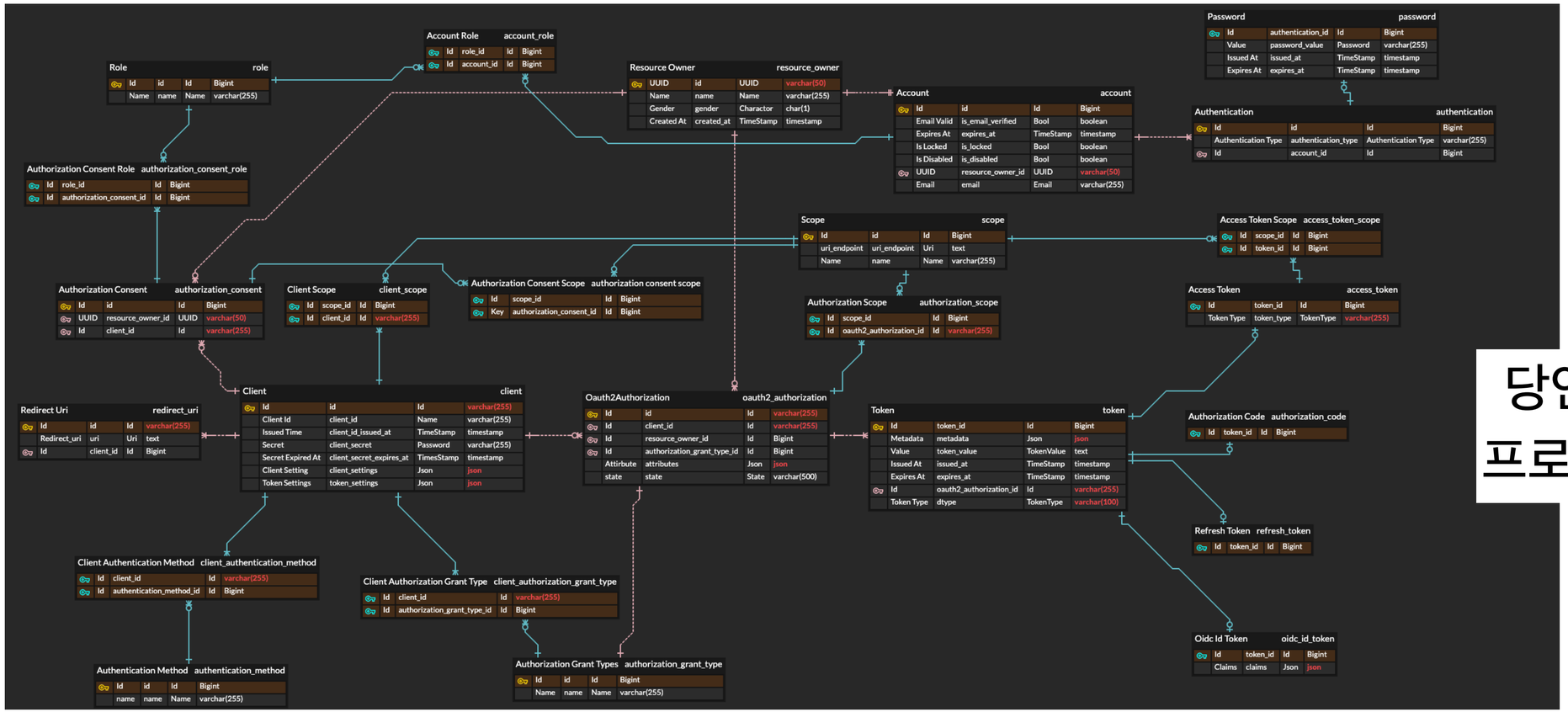
Cypher Query Language

예시로 이해하는 Graph Database의 강력함

20180237 김온지

왜 RDB안쓰고?

관계형 데이터 베이스는.. 이름 값을 못한다.



단점들..

- 너무 많은 매핑 테이블
- 유연하지 못한 스키마(변경도 힘듬)
- Null 값이 많을 경우 비효율적
- 관계를 탐색할때 매번 JOIN 발생
- 연속적인 관계 탐색이 힘들다

당연히 안좋다는건 절대 아님
프로젝트 요구사항에 따라 선택

SQL Structured Query Language

Cyber Query Language

Neo4j의 쿼리 언어

Basic Query

이름이 Tom Hanks 인 사람 찾아줘


```
1  MATCH (tom:Person {name: 'Tom Hanks'})
2  RETURN tom
```

```
1  MATCH (tom:Person)
2  WHERE tom.name = “Tom Hanks”
3  RETURN tom
```

Basic Query

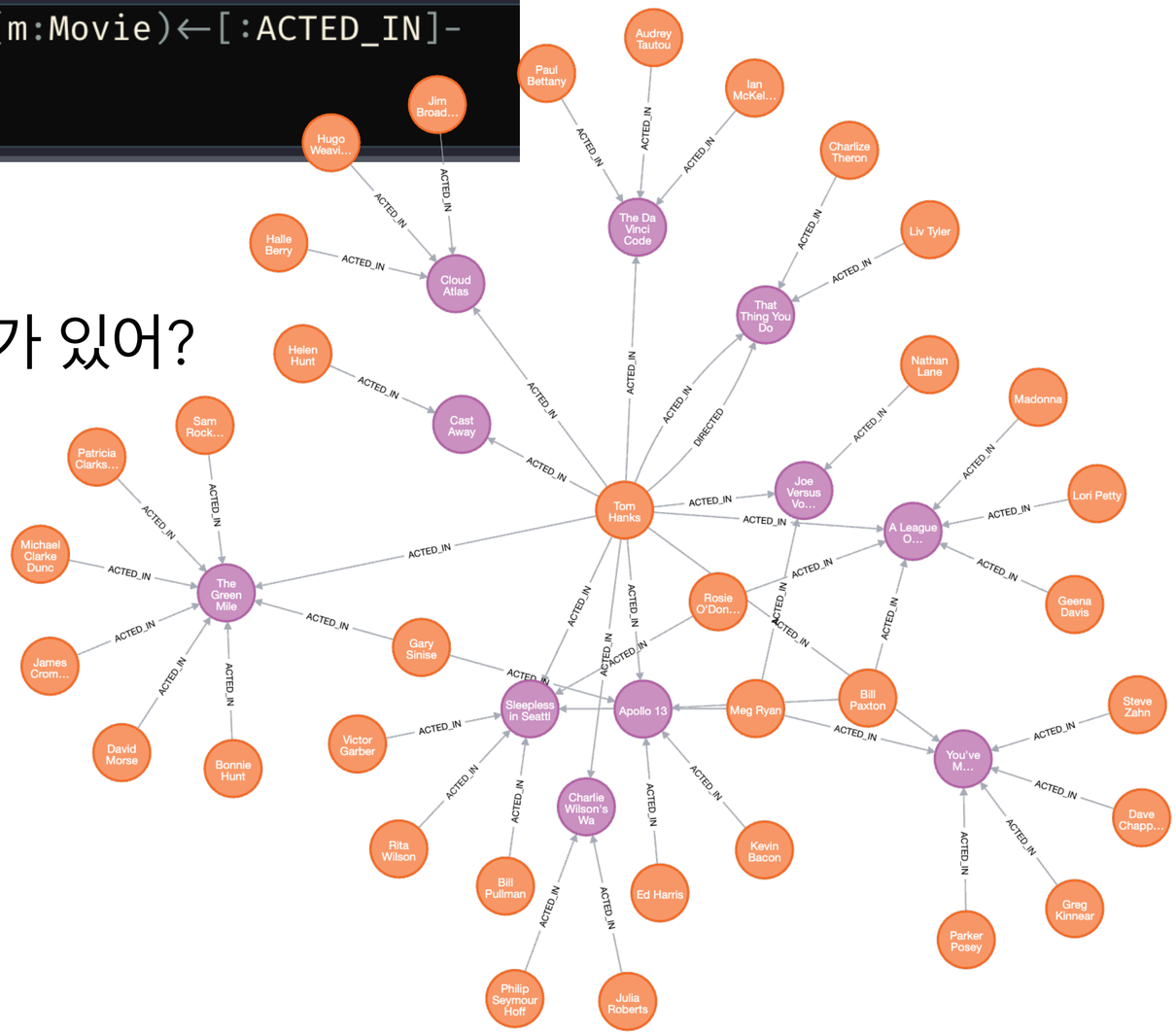
Tom Hanks 참여한 영화는 뭐가 있어?

```
1  MATCH (tom:Person {name: 'Tom Hanks'})-[r:ACTED_IN]→(movie:Movie)
2  RETURN tom, r, movie
```

Basic Query

```
1  MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]→(m:Movie)←[:ACTED_IN]-
   (coActor:Person)
2  RETURN coActor, tom, m
```

Tom Hanks와 같은 영화에서 연기한 배우는 누가 있어?



Basic Query

Tom Hanks와 같은 영화에서 연기한 배우 “이름” 좀 알려줘

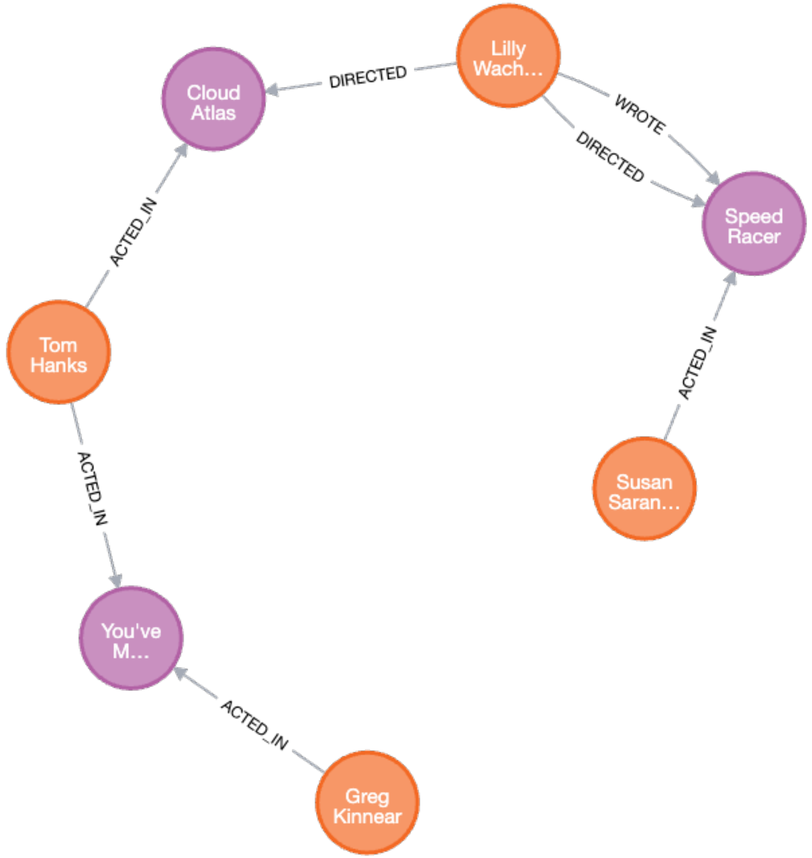
```
1 MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]→(:Movie)←[:ACTED_IN]-
  (coActor:Person)
2 RETURN coActor.name
```

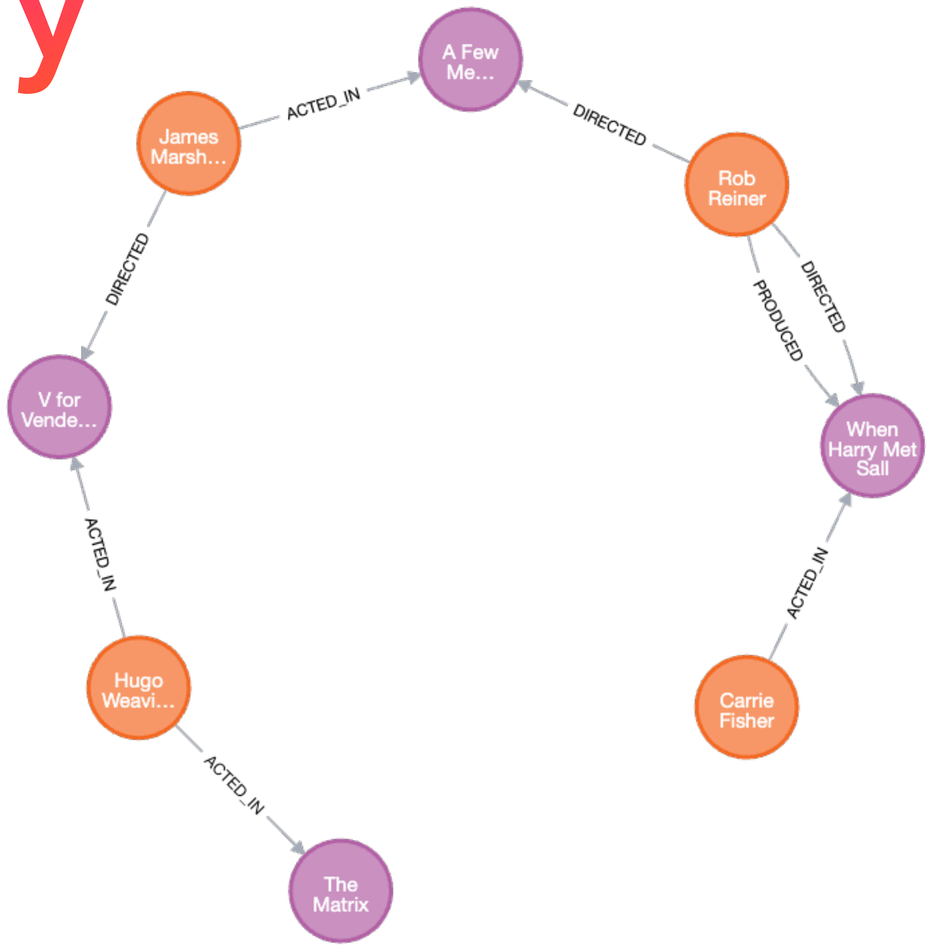
coActor.name
"Ed Harris"
"Gary Sinise"
"Kevin Bacon"
"Bill Paxton"
"Parker Posey"
"Greg Kinnear"
"Meg Ryan"
"Steve Zahn"
"Dave Chappelle"

Basic Query

Susan과 Greg Kinnear 사이에 가장 짧은 경로가 뭐야?

```
1  MATCH path = shortestPath((p1:Person)-[*1..10]-(p2:Person))
2  WHERE p1.name = "Susan Sarandon" AND p2.name = "Greg Kinnear"
3  RETURN path
```





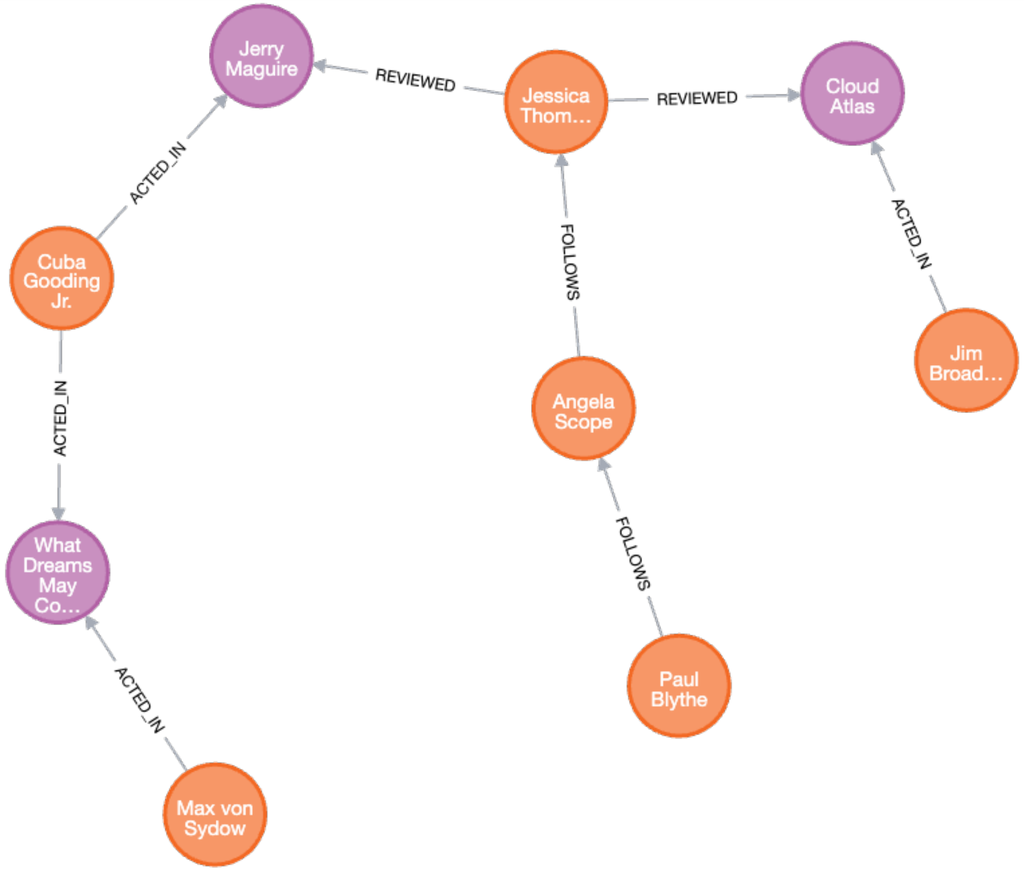

```
1  MATCH path = shortestPath((p:Person)-[*1..10]-(m:Movie))
2  WHERE p.name = "Carrie Fisher" AND m.title = "The Matrix"
3  RETURN path
```

Carrie Fisher하고 영화 Matrix 사이에 무슨 관련이 있어?

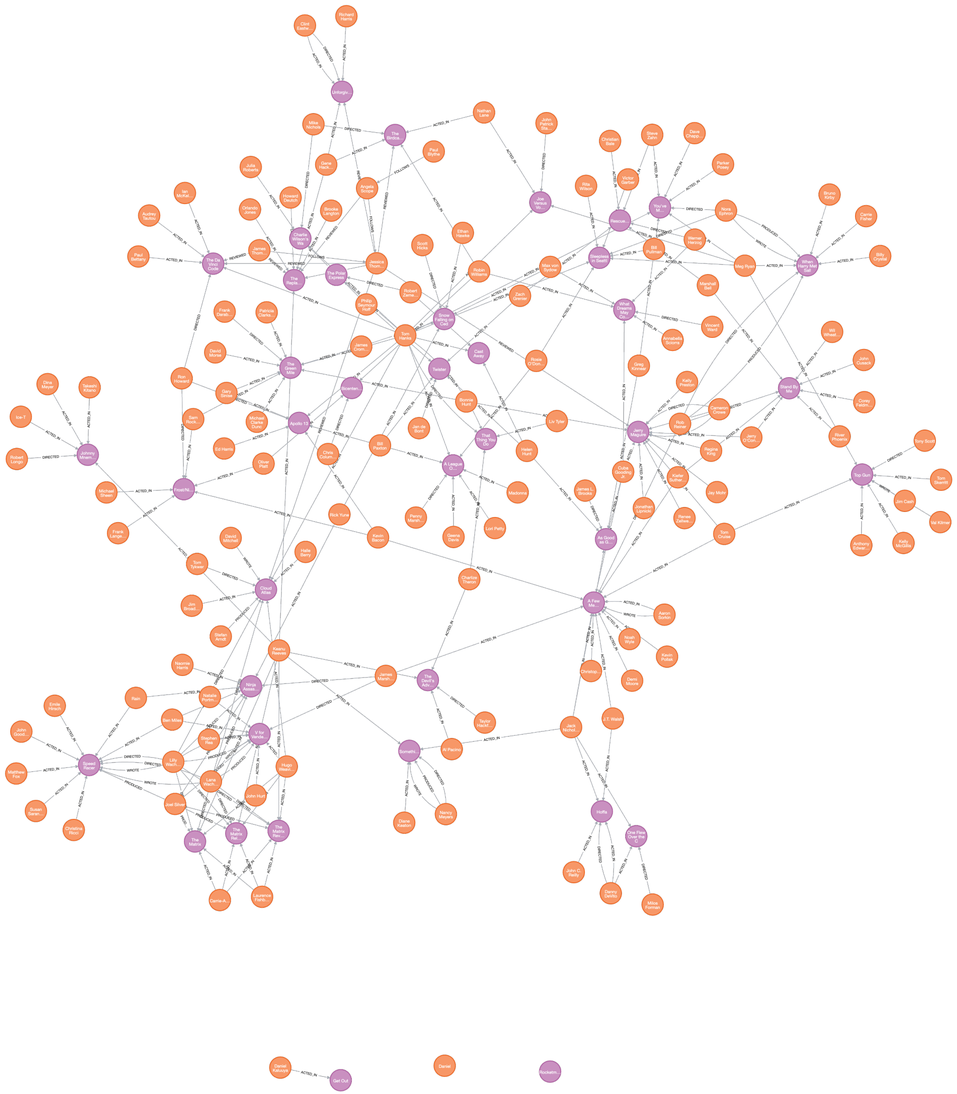
Basic Query

```
1  MATCH path = shortestPath((p1:Person)-[*..10]-(p2:Person))
2  WHERE p1.name = "Jim Broadbent" AND
3      (p2.name = "Max von Sydow" OR p2.name = "Paul Blythe")
4  RETURN path
```

Jim하고



Basic Query



MATCH (n) RETURN n

나한테까지 코로나가 전파된 경로

나와 내 친구와 코로나 전파의 공통 뿌리 찾기

사기 탐지

마케팅 성과 분석

네비게이션

웹 하이퍼링크의 관계 파악

특정 후보를 지지하는 여성의 주요 요인 파악

군집 분석

추천 Query 예시

Tom Hanks과 한다리 건너 영화를 찍은 사람들을 추천해줘

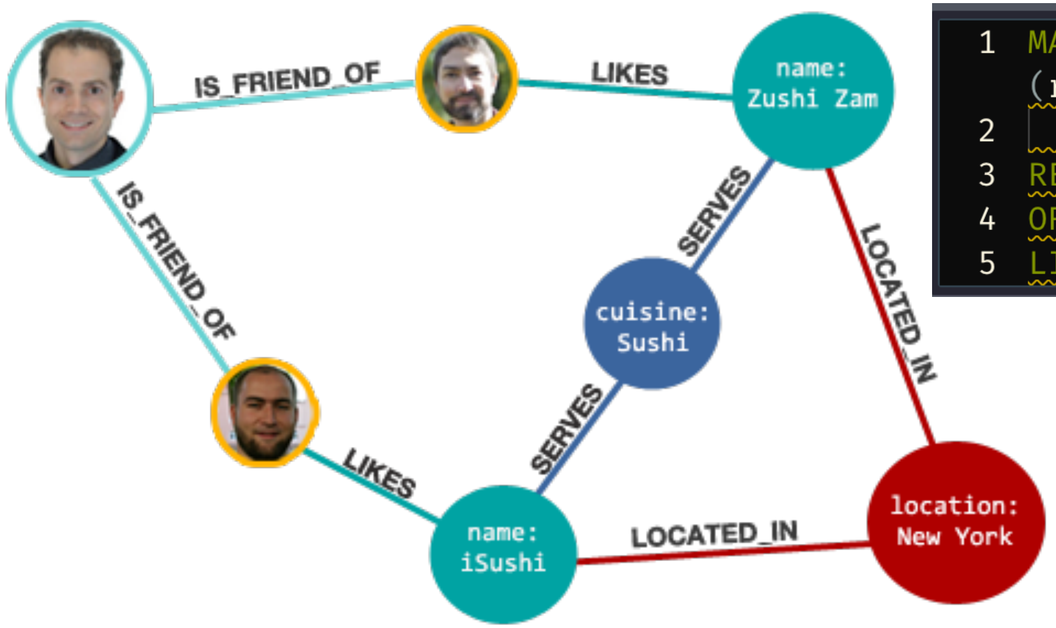
```
1  MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]→(movie1:Movie)←[:ACTED_IN]-
   (coActor:Person)-[:ACTED_IN]→(movie2:Movie)←[:ACTED_IN]-(coCoActor:Person)
2  WHERE tom <> coCoActor
3  AND NOT (tom)-[:ACTED_IN]→(:Movie)←[:ACTED_IN]-(coCoActor)
4  RETURN coCoActor.name, count(coCoActor) as frequency
5  ORDER BY frequency DESC
6  LIMIT 5
```

Table | Text | Code

	coCoActor.name	frequency
1	"Tom Cruise"	5
2	"Zach Grenier"	5
3	"Cuba Gooding Jr."	4
4	"Keanu Reeves"	4
5	"Jack Nicholson"	3

추천 Query 예시

내 친구들이 좋아하는, New Yourk에 있는 Sushi 레스토랑은 뭐가 있어?




```
MATCH (person:Person {name: 'Philip'})-[:IS_FRIEND_OF]→(friend)-[:LIKES]→
(restaurant:Restaurant)-[:LOCATED_IN]→(loc:Location {location: 'New York'}),
      (restaurant)-[:SERVES]→(type:Cuisine {type: 'Sushi'})
RETURN restaurant.name, count(*) AS occurrence
ORDER BY occurrence DESC
LIMIT 5
```

추천 Query 예시

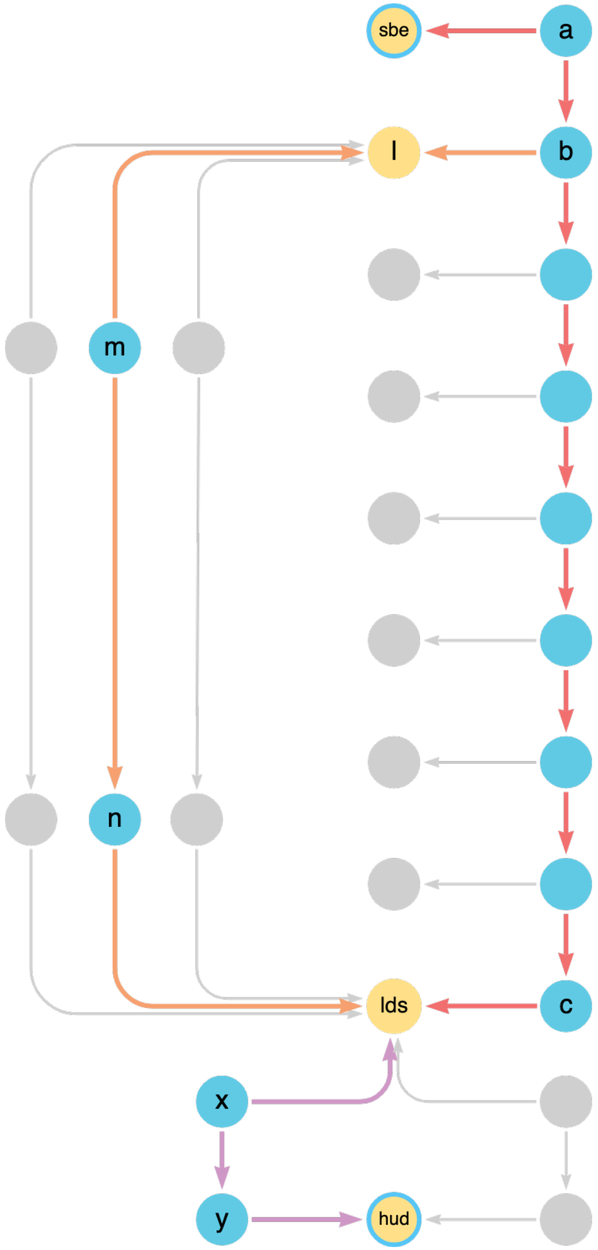
이 레시피랑 가장 유사한(비슷한 재료를 많이쓰는) 레시피는 뭐가 있어?

이 요리랑 비슷한 요리 추천해줘

```
1  MATCH (r:Recipe {id:'97123'})-[:CONTAINS_INGREDIENT]→(i:Ingredient)←
   [:CONTAINS_INGREDIENT]-(rec:Recipe)
2  WITH rec, COUNT(*) as commonIngredients
3  RETURN rec.name, rec.id ORDER BY commonIngredients DESC LIMIT 10
```

"Black Forest sundaes with brownies"	"100341"
"Christmas pud cupcakes"	"101166"
"Little Black Forest cakes"	"103089"
"Chocolate & almond puds with boozy hot chocolate sauce"	"97938"
"Chocolate & Earl Grey torte"	"98097"
"Lighter Chocolate cake with chocolate icing"	"5020831"
"Sunken drunken chocolate cake"	"96335"
"Espresso, chocolate & chilli cake with coffee cream"	"4542501"
"Orange chocolate tart"	"94802"
"Chocolate & caramel ombre cake"	"102539"

복잡한 쿼리 예시



Starbeck - Leeds - Huddersfield 길찾기

```
MATCH (:Station {name: 'Starbeck'})<-[:CALLS_AT]-
       (a:Stop {departs: time('11:11')})-[:NEXT]->*(b)-[:NEXT]->*
       (c:Stop)-[:CALLS_AT]->(lds:Station {name: 'Leeds'}),
      (b)-[:CALLS_AT]->(l:Station)<-[:CALLS_AT]-(m:Stop)-[:NEXT]->*
       (n:Stop)-[:CALLS_AT]->(lds),
      (lds)<-[:CALLS_AT]-(x:Stop)-[:NEXT]->*(y:Stop)-[:CALLS_AT]->
       (:Station {name: 'Huddersfield'})
WHERE b.arrives < m.departs AND n.arrives < x.departs
RETURN a.departs AS departs,
       l.name AS changeAt,
       m.departs AS changeDeparts,
       y.arrives AS arrives
ORDER BY y.arrives LIMIT 1
```

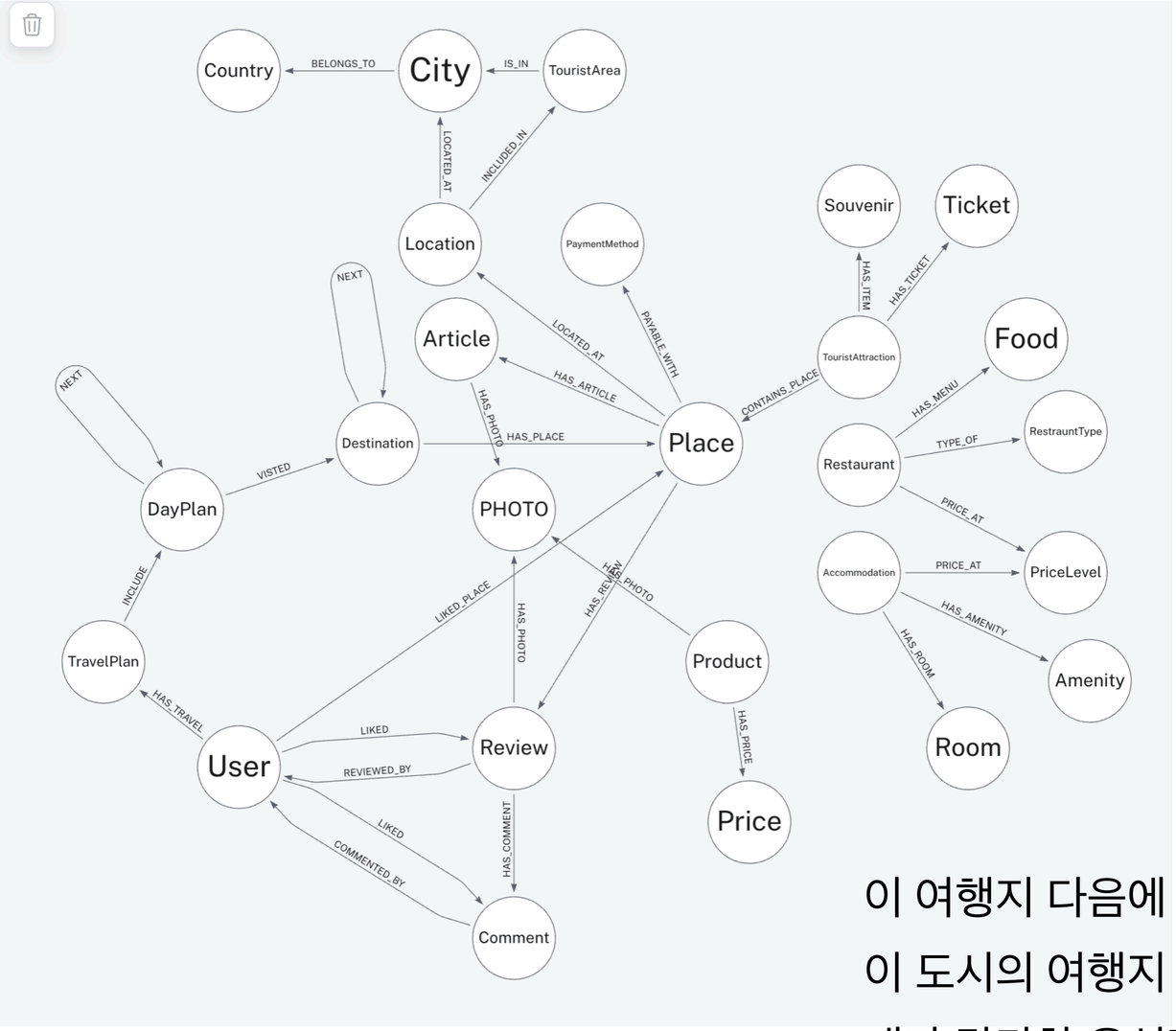
Table 13. Result

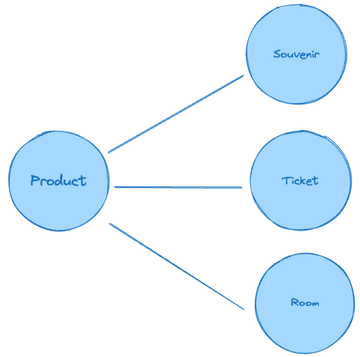
departs	changeAt	changeDeparts	arrives
"11:11:00Z"	"Harrogate"	"11:20:00Z"	"12:07:00Z"

Rows: 1

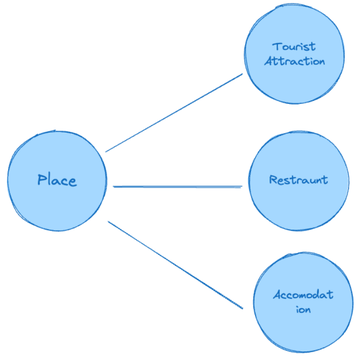
https://neo4j.com/docs/cypher-manual/current/patterns/concepts/

내가 설계한 Graph Model





IS-A Relation



IS-A Relation

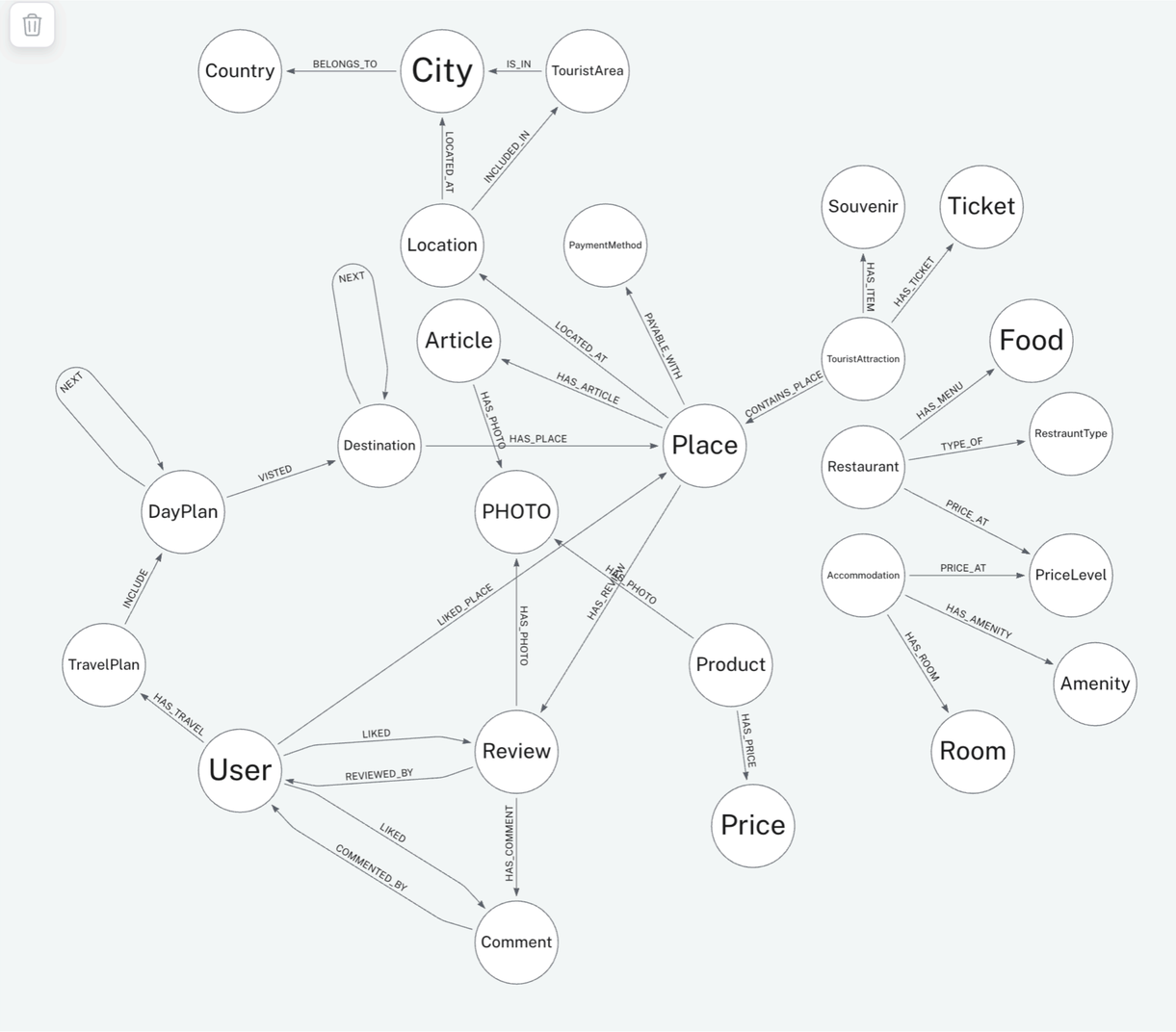
실제로는 Multiple Tag로

이 여행지 다음에 많이간 여행지

이 도시의 여행지 사진

내가 갈만한 음식점(기존 기록, 친구 기록)

내가 설계한 Graph Model



Place.graphqls | Neo4jConfig.java

```
    tips: [String!]
    openingHours: [Period!]
    tags: [String!] # 검색에 활용됨
    thumbnail: Photo
    photos: [Photo!]
    # 쿼리형 필드
    availablePaymentMethods: [PaymentMethod!] # 결제 수단
    userLikeCount: NonNegativeInt!
    userRatingCount: NonNegativeInt!
    averageRating: NonNegativeFloat # 0.0 ~ 5.0
    reviews: [Review!]!
    relatedPlace: [Place!] # GDB
    nearPlaces: [Place!] # GDB or es
}

# TouristAttraction : 관광지
type TouristAttraction implements Place {
    id: ID!
    title: String!
    localTitle: String
    editorialSummary: String
    location: Location!
    content: Article
    phoneNumber: String
    websiteUri: String
    businessStatus: BusinessStatus!
    directions: String # 오시는 길
    tips: [String!]
    openingHours: [Period!]
    tags: [String!] # 검색에 활용됨
    # 쿼리형 필드
    thumbnail: Photo
    photos: [Photo!]
    availablePaymentMethods: [PaymentMethod!] # 결제 수단
```

감사합니다.